ATTACHMENT 10

UNITED STATES DISTRICT COURT
NORTHERN DISTRICT OF CALIFORNIA
SAN JOSE DIVISION

CISCO SYSTEMS, INC., Plaintiff, v. ARISTA NETWORKS, INC., Defendant.	Case No. 5:14-cv-05344-BLF (PSG)

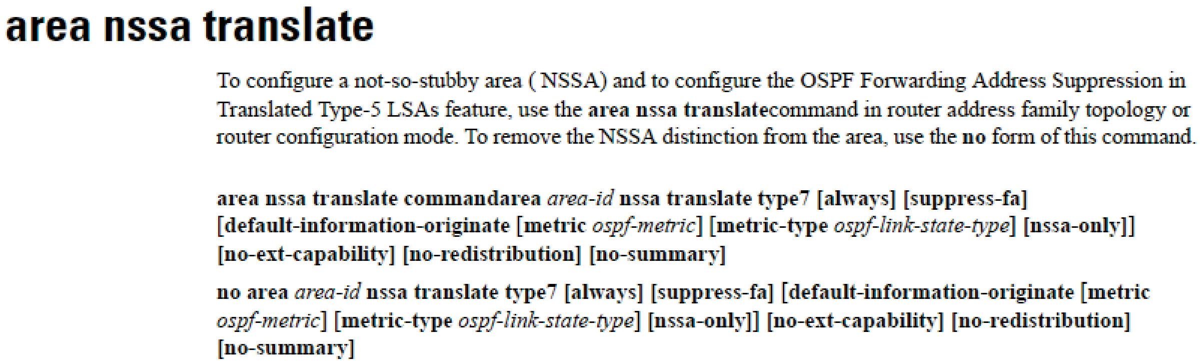
EXPERT REPORT OF JOHN R. BLACK, JR.

June 3, 2016

CONTAINS HIGHLY CONFIDENTIAL MATERIAL
SUBJECT TO PROTECTIVE ORDER

John R. Black, Jr.

appear in the form Cisco is asserting in this case. There are descriptions of commands, but these are typically explanations of how to construct different commands and what syntax they follow. For example, the asserted CLI command “area nssa translate type7 always” appears in Cisco manuals as follows, with both typographical errors as well as optional parameters shown in brackets:



area nssa translate

To configure a not-so-stubby area (NSSA) and to configure the OSPF Forwarding Address Suppression in Translated Type-5 LSAs feature, use the **area nssa translate**command in router address family topology or router configuration mode. To remove the NSSA distinction from the area, use the **no** form of this command.

area nssa translate commandarea *area-id* **nssa translate type7 [always] [suppress-fa] [default-information-originate [metric** *ospf-metric*] **[metric-type** *ospf-link-state-type*] **[nssa-only]] [no-ext-capability] [no-redistribution] [no-summary]**

no area *area-id* **nssa translate type7 [always] [suppress-fa] [default-information-originate [metric** *ospf-metric*] **[metric-type** *ospf-link-state-type*] **[nssa-only]] [no-ext-capability] [no-redistribution] [no-summary]**

CSI-CLI-00296282 (“Cisco IOS IP Routing: OSPF Command Reference”) (Cisco IOS 15.4).

522. To my knowledge, Cisco does not publish a list similar to Exhibit 1 to the complaint. Other non-literal aspects of Cisco’s code that are allegedly copied are the abstract hierarchy of certain commands, the fact that EOS operates under modes similar to those found in versions of Cisco operating systems, and that there are some similarities in formatting of some responses that are printed on a screen in responses to certain commands. I have considered all of these non-literal aspects of Cisco’s asserted works and am of the opinion that these aspects comprise (together with the other recognized CLI commands, arguments, parameters and other command responses not asserted) a method of operation for controlling a Cisco device. Individually, some of these aspects of the Cisco CLI are nothing more than a concept or a system. There are several reasons

why these aspects comprise a system or a method of operation.

523. Each of the asserted CLI commands listed in Exhibit 1 to the Complaint serve the sole purpose of enabling a network administrator to adjust a setting, reveal a status, or enable a feature on the corresponding network device. In essence, these commands are nothing more than a means to invoke a functionality, much like a physical knob on older vintage electronics.[99] Indeed, many in the networking industry refer to the settings that one can access via CLI commands as “knobs.” Taken together, the asserted CLI commands, together with all of the other CLI commands, constitute a group of controls to administrator to manage, operate, and report on the status of the device. In this sense, the sum total of the commands recognized by any Cisco operating system (or any operating system) are the controls that unlock the functionality of that operating system or its associated hardware. This set of controls is, in my opinion, a system or a method of operation.

524. As further demonstration of this fact, the asserted CLI commands (or fragments), as a rule, do not appear as they are listed in Exhibit 1 within Arista’s source code. Thus, Cisco appears to base its contention that Arista’s EOS software infringes Cisco’s copyrights on the assertion that Arista designed the EOS parser to perform the *function* of parsing certain text strings that are also parsed by Cisco’s IOS parser, even though Arista wrote its parser code from scratch, and without reference to Cisco’s parser code. This is a purely functional relationship--further confirming that the asserted CLI commands are a method of operation (i.e., they are something that the accused software recognizes according to one of its functions, generally not part of the accused software

[99] Terry Slattery described Cisco’s CLI as the “mechanism for how their customers access control and configure and manage their devices.” Slattery Dep. Tr. at 149.

itself).

525. Considering this question at another level, the way the commands are parsed confirms, in my opinion, that they are a method of operation. In both Arista's parser and Cisco's parser[100] the commands are ingested by examining characters until a space appears in the character string. That set of characters is either determined to be a valid keyword, or argument or parameter, or not. The process repeats for the next set of characters entered before the next space, and so on, until the last character is entered. As such, each word in a given command serves merely to invoke a set of possible functions that are available in the code, until by the end of a valid command there is only one unambiguous function to perform. In this way, the asserted CLI command is akin to a mailing address: by parsing country, zip, state, city, street and apartment number one reaches an unique destination; the elements of the address that lead you there is a method of addressing. Likewise the command words that constitute a command specify a unique function to perform in the device, and thus the commands themselves are merely a method of operating the network device.

526. Another aspect of parsing of the CLI commands supports my conclusion that the asserted aspects of Cisco's CLI constitute a method of operation. In practice, it is rare for a network administrator to enter the full character string of a command word. These command words become so familiar to network administrators that they take advantage of any possible shortcut to typing out multi-word commands. Much more often the network administrator will only enter the first few characters of each command word and allow the software to "auto-complete" so long as there is only one valid

[100] This claim relates to the source code versions of the respective parsers I have seen. That is, Arista EOS 4.15 and Cisco IOS 12.4.

of allegedly copied protected expression are subsets of these screen displays--sometimes a phrase or a single line of text, sometimes a series of abbreviations, sometimes a series of data headings, etc. In all cases, however, the excerpts are part of an overall system or method of operation, namely obtaining and organizing data regarding state.

540. Consider for example the list of abbreviations for which Cisco claims copyright protection.[105] This is a purely functional recitation of common industry protocols or terms associated with those protocols. Such a list uses the first word of the term(s) as abbreviations for the standard term. Neither the standard term, nor the idea of abbreviating the term, nor using the first letter for the abbreviation, is anything more than a system for coding the “show ip route” status.[106]

541. Consider also, for example, Cisco’s claim of copyright infringement based upon the similarity of parameters identified in response to the “show snmp” command.[107] SNMP is a standardized protocol, and I understand that Cisco is not claiming any proprietary rights in SNMP in particular. The parameters listed here are functional elements available in an SNMP implementation. These short descriptions of the parameters are purely functional. These descriptions are inseparable from the underlying system that allows for the tracking of each such metric. Moreover, it would be be expected that customer scripts would seek these descriptions in order to programmatically read the state of the SNMP operations. In this way, the allegedly copied descriptions form a systematic roadmap for scripts to determine the state of the switch automatically by relying on these expected identifiers of different metrics and

[105] *Id.* at 1.
[106] I note also that Arista has not used similar abbreviations for features it does not use, such as Cisco’s proprietary EIGRP protocol.
[107] *Id.* at 2.

data.

542. Finally, consider as another example the headings for "show port-security" in which Cisco claims copyright.[108] Again, these headings represent a system of organizing and reciting network status information. These headings are similar to a blank form. By using these headings to display network state data, Arista assures that such data appears in the columns expected by network administrators, as one might expect a certain layout for the standings table of a sports league or the layout of contact information for a phone list. Additionally, as with the SNMP parameters, these headings are the type of screen displays that would likely be used by network scripts to locate automatically relevant state information.

B. The Asserted Aspects of the Cisco CLI Lack Originality

1. The Asserted CLI Commands are not original.

543. Based on my analysis below and in this Report, it is my opinion that the asserted CLI commands consist primarily of terms taken from (and defined and used in) published industry standards from standards setting bodies like the IETF and IEEE, from documents like Informational RFCs that describe *de facto* industry practices or industry best practices, and/or widely and commonly used industry terms relating to the CLI command's functionality. As described below, and based on my review of deposition testimony (in particular, the testimony of Cisco engineers Tong Liu, Devadas Patil, Anthony Li, Kirk Lougheed, Abhay Roy, Ram Kavasseri, and Pradeep Kathail) and documents produced in this lawsuit, the individuals responsible for adding new CLI commands to a Cisco product are typically very familiar with, have access to, and

[108] *Id.*

directly consult any industry standards (for example, IETF RFCs and IEEE standards) that might relate to those commands before such commands are added to Cisco's CLIs. Those individuals, based on their education and training leading up to their roles as Cisco engineers, are also typically familiar with widely used networking terminology before they add any new CLI commands to Cisco's CLIs.

544. A summary of the relevant IETF and IEEE standards and related documents (including non-Standards Track RFCs and Internet Drafts that describe *de facto* industry standard practices and protocols) is attached to this Report as **Appendices A and B**. Those appendices identify the many IETF and IEEE documents that describe, define, and use many of the command keywords found within the asserted CLI commands (and command fragments), and also discuss several common command keywords widely used in the industry. **Appendices E** through **H** also highlight the widely used and familiar CLI commands and keywords that are known to and used by many vendors in the networking industry.

545. It is also my opinion that several of the asserted CLI commands, and aspects of their syntax, were copied from networking systems and products that predated the addition of those CLI commands (or CLI syntax conventions) to Cisco's CLIs. Based on my review of deposition testimony and documents produced in this lawsuit, many aspects of the asserted CLI commands--and some entire commands--were taken from Digital Equipment Corporation (DEC) products, including DECSYSTEM-20, TOPS-20, and DEC VAX/VMS products. Cisco admits Cisco viewed DEC's CLI as a "defacto industry standard" and copied "some of the common commands".[109]

[109] Email from Nicholas Thille to Robert Snyder, 11-May-1994 [CSI-CLI-04978736]

546. Moreover, as described below, those systems--and other legacy systems including UNIX, MS-DOS, and others--featured CLIs with many of the same basic features and functionality (including command modes and prompts) that Cisco asserts in this lawsuit. Based on my review of deposition transcripts and documents produced in this lawsuit, several Cisco engineers responsible for adding those features to the Cisco IOS CLI had access to, and were already familiar with, the same or similar features in DEC, UNIX, and other legacy systems.

2. Asserted aspects of the Cisco CLI came from prior legacy systems, and did not originate from Cisco.

547. Based on my analysis below and in this Report, it is my opinion that several fundamental aspects of the IOS CLI that Cisco is asserting against Arista were copied and taken directly from other CLIs that existed in non-Cisco systems and software.

548. Many features of the Cisco IOS CLI, including features that Cisco asserts copyright over in this lawsuit, originated from a non-Cisco operating system called TOPS-20. TOPS-20 was not developed by Cisco, but was instead developed by a company called Digital Equipment Corporation, or DEC. *See, e.g.*, Lougheed Dep. Tr. at 43; *see also* ARISTANDCA00038298 (TOPS-20 Commands Reference Manual, AA-FP65A-TM, Sept. 1985);

549. TOPS-20 was the operating system for DEC equipment, including the DECSYSTEM-20, which have been in use since the 1970s, as confirmed by Kirk Lougheed, who worked on DECSYSTEM-20 systems in the early 1980s before he began any work on the software that would become the Cisco IOS software. *Id.* at 33, 43.

550. For example, TOPS-20 supported different modes where a given user had different privileges and capabilities. *See, e.g.*, Lougheed Dep. Tr. at 256-259. As Mr. Lougheed confirmed at his deposition, TOPS-20 had a “human interface” program called EXEC, similar to the “User EXEC” command mode that is used in Cisco IOS and in almost all CLIs used in the networking industry today. *Id.* at 50-51 (“**Q.** Are you familiar with something called EXEC, E-X-E-C, in TOPS-20? **A.** Yes **Q.** What is that? **A.** It's a program. **Q.** What does it do? **A.** It's the interface -- the human interface for the TOPS-20 operating system. … **Q.** Was the EXEC program part of TOPS-20? **A.** Yes.”); *see also id.* at 381-382 (confirming that “Exec” and “User EXEC” refer to the same command mode); *see also* ARISTANDCA00038298 at Page Intro-9 (TOPS-20 Commands Reference Manual, AA-FP65A-TM, Sept. 1985) (discussing the “EXEC” TOPS-20 Command Processor).

551. In addition to the EXEC, TOPS-20 had a “privileged” mode. *See, e.g.*, Lougheed Dep. Tr. at 55-56 (“**Q.** Do you understand whether there was a privileged mode in TOPS-20? **A.** Yes. There is -- there is a privileged mode. **…** **Q.** Did privileged mode allow you a different set of commands than in a non-privileged mode? **A.** There were a -- yes.”). Like Cisco IOS and the many different CLIs I analyze in this Report, TOPS-20 indicated to the user that they were in different modes by changing the command prompt. *Id.*

552. To get into “privileged” mode (as Mr. Lougheed characterized it) in TOPS-20, the user would enter the command “enable” to access that mode. Lougheed Dep. Tr. at 56; ARISTANDCA00038298 at Page 137 (TOPS-20 Commands Reference Manual, AA-FP65A-TM, Sept. 1985) (discussing the “enable” command).

553. This standard CLI functionality used in TOPS-20--entering the "enable" command to move from "User EXEC" mode to "PRIVILEGED EXEC" mode--is also present in most of the CLIs from the various networking vendors analyzed in this Report, including in the Cisco IOS CLI. *See*, *e.g.*, CSI-CLI-00024968 (Cisco Configuration Fundamentals, Config. Guide, Cisco IOS Release 15.2T) (showing that entering "enable" at the User EXEC command prompt "[e]nables privileged EXEC mode").[110]

554. This separation of different users (or accounts) with different system privileges is also found in legacy UNIX systems that pre-date Cisco IOS. Those UNIX systems feature a "root user" or "superuser" that had more privileges than other accounts. *Id.* Importantly, most UNIX system implementations use the command prompt "#" to indicate that the root user account is logged in. This is equivalent to the "#" prompt used in the privileged mode of the Stanford SUMEX CLI, the "#" prompt used for the privileged mode in the Stanford EECF software CLI, and the "#" prompt used for privileged EXEC modes used in almost all of the switch and router CLIs analyzed in this Report.

555. UNIX also supported a "clear" command, as Mr. Lougheed admitted at his deposition. See Lougheed Dep. Tr. at 173-174. While Mr. Lougheed testified that the "clear" command in UNIX was not used "in the sense that the Cisco CLI uses 'clear'" in its CLI commands, I note that some of the Arista CLI commands that Cisco accuses in this lawsuit are similarly not used in the same way that the Cisco CLI uses them (*e.g.*, "ip

[110] Elsewhere in this Report I discuss the various command modes supported by many different vendors in the networking industry. On those same cited pages, most of the vendors use the "enable" command to move from a non-privileged command mode to a privileged command mode. *See, e.g.*, 19006JNPR00130657 at Page 194 (Juniper JUNOEs Internet Software - Command Reference A-M 7.1x) ("From User Exec mode, [the 'enable' command] enters Privileged Exec mode at the specified privilege level."); HPE44508 at Page 29 (HP Basic Operation Guide, 2013) (describing the "enable" command to enter a privileged "Manager" mode).

protocol" and "max-connections," which are used for the "Monitor Reachability" and LANZ features found only in Arista switches). Moreover, using "clear" as a command long predates its use in Lougheed's work: I used to type "clear" on my TRS-80 as early as 1978 when I wanted to erase variables from memory[111].

556. TOPS-20 also supported the use of several common CLI features discussed throughout this report. For example, the TOPS-20 CLI supported command abbreviations being entered into the CLI, just like almost every other CLI analyzed in this Report. *See* ARISTANDCA00038298 at Page Intro-5 (TOPS-20 Commands Reference Manual, AA-FP65A-TM, Sept. 1985) (discussing the "abbreviation" feature where "[t]he smallest unique abbreviation for a command or argument will stand for the entire word"). TOPS-20 also supported command completion, where it would automatically complete an abbreviated command word when the user hit a particular key (ESC in the case of TOPS-20). *Id.* at INTRO-6; *see also* ARISTANDCA00009216 at Section 1.1, Table 1-1 (TOPS-20 User's Guide, 1988) (discussion command completion functionality).

557. And TOPS-20 also supported a context-sensitive help system triggered by the use of a question mark "?" character in the CLI, just like almost every other CLI analyzed in this Report. *See* ARISTANDCA00038298 at Page Intro-6 (TOPS-20 Commands Reference Manual, AA-FP65A-TM, Sept. 1985) ("Whenever you type a question mark (?) instead of (or even in the middle of) a TOPS-20 command or command argument, the system responds with instructions or a list of possible completions. By using questions marks and recognition when you are unsure of the proper command or argument, you can have the system help you during your terminal session.").

[111] See for example, the TRS-80 manual at http://www.classiccmp.org/dunfield/kyocera/m100qr.pdf

558. Former Cisco employee Anthony Li confirmed these features of TOPS-20 in his deposition: "TOPS-20 had a command syntax that was somewhat similar to [DEC VAX] VMS [discussed further below]. The notable difference was that TOPS-20 allowed for a command completion, and so you could use escape and tab and question mark characters to interact directly with the command line interpreter while you were typing a command line." Li Dep. Tr. at 21-24 (discussing his use of TOPS-20 in 1982, before Cisco was created and confirming his "hands-on" experience with TOPS-20).

559. Former Cisco employee Gregory Satz also testified about the similarities between TOPS-20 and the Cisco IOS CLI. For example, Mr. Satz confirmed that the Stanford TIP (Terminal Interface Processor) software upon which the Cisco IOS software was based "mimicked the TOPS-20 style of parsing, and it -- there were commands that people would use to just have the device do what it does day-to-day, and there were commands that administrators or users who needed to maintain the device in the network would use. And so privilege commands were the latter set, and TOPS-20 had a very similar model." Satz Dep. Tr. at 26-30; *see also id.* at 47-49 (discussing the EXEC mode in TOPS-20 the use of an "enable' command to enter a privileged mode). And Mr. Satz also confirmed that the use of "show" commands was not unique or original to Cisco:

> **Q.** Had you ever heard of or used show commands in any context before you went to Cisco?
>
> **A.** Every computer has show commands. I mean every operating system had used the word "show" as a way to convey internal information outward.

Id. at 32; *see also* ARISTANDCA13228395 (TOPS-20 DECnet-20 Programmer's Guide and Operations Manual AA-50918-TM Jan. 1980) (listing "show" and "set" commands as part of the Network Control Program, or NCP, commands).

560. DEC also developed an operating system for its VAX line of products called VMS. VAX means virtual address extension and VMS is virtual memory system. *See* Anthony Li Dep. Tr. at 17-21 (discussing his experience with the DECnet network at USC in the 1980s and DEC VAX/VMS operating system, and noting that his first exposure to VAX/VMW was in 1981, before Cisco was created). Like TOPS-20, the VAX/VMW operating system had a command line interface, and used commands like "SHOW" and "SET". *Id.* at 20. As former Cisco employee Anthony Li explained at his deposition, the VAX/VMS operating system--which predated Cisco IOS--was a "very standard command-and-response interface." *Id.*; *see also* ARISTANDCA13228395 (TOPS-20 DECnet-20 Programmer's Guide and Operations Manual AA-50918-TM Jan. 1980) (listing "show" and "set" commands as part of the Network Control Program, or NCP, commands).

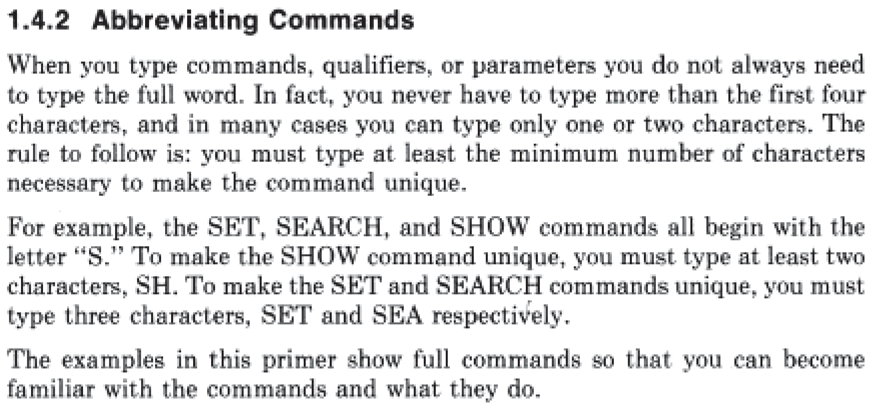
561. As Mr. Li further explained, the entire operating system CLI for DEC VAX/VMS was built around DCL, or Digital Command Language. Li Dep. Tr. at 21. The "show" commands supported by DEC VAX/VMS as early as 1981 followed a hierarchical command syntax very similar to the "show" commands disputed in this lawsuit:

> **Q.** Did the show commands in VAX/VMS follow any particular syntax?
>
> **A.** Yes. They typically were invoked by show and then usually an object name and then a set of parameters.

> … The set commands were pretty much the same way.

Id. at 21-22. Mr. Li's description of the DEC VAX/VMS command syntax, particularly with respect to "show" and "set" commands, is consistent with my experience using VAX/VMS, as well as the command reference manuals describing the VAX/VMS commands. *See* ARISTANDCA13229400 (VAX/VMS Primer Order No. AA-0030C-TE May 1982).

562. Indeed, the VAX/VMS manual also confirms that its CLI supported the widely used command abbreviation functionality, in addition to "show" and "set" commands:



1.4.2 Abbreviating Commands

When you type commands, qualifiers, or parameters you do not always need to type the full word. In fact, you never have to type more than the first four characters, and in many cases you can type only one or two characters. The rule to follow is: you must type at least the minimum number of characters necessary to make the command unique.

For example, the SET, SEARCH, and SHOW commands all begin with the letter "S." To make the SHOW command unique, you must type at least two characters, SH. To make the SET and SEARCH commands unique, you must type three characters, SET and SEA respectively.

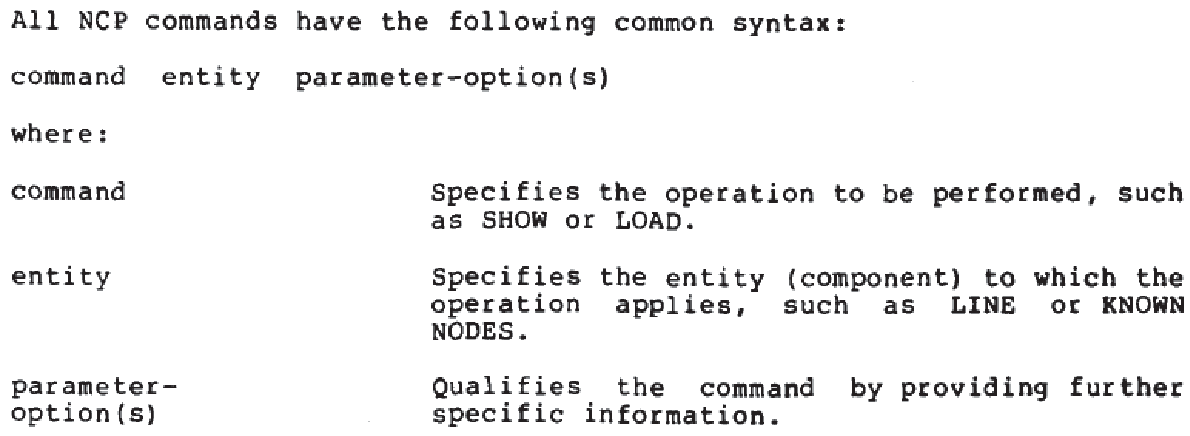
The examples in this primer show full commands so that you can become familiar with the commands and what they do.

See ARISTANDCA13229400 at Page 1-6 (VAX/VMS Primer Order No. AA-0030C-TE May 1982).

563. DEC also developed what it called its DECnet Digital Network Architecture, which was released in several phrases. DECnet Phase III was released in 1980, well before Cisco existed. *See* ARISTANDCA13228703 (DECnet DIGITAL Network Architecture (Phase III) Network Management Functional Specification Order No. AA-K181 A-TK Version 2.0.0 October 1980). Phase IV was released in 1983, also before Cisco was created. *See* ARISTANDCA13229492 (DECnet Digital Network

Architecture Phase IV Network Management Functional Specification Order No. AA-X437A-TK Dec. 1983). Phase V was released in 1987. *See* KL-00000251 (DECnet DIGITAL Network Architecture (Phase V) General Description Order No. EK-DNAPV-GD Sept. 1987, produced from Kirk Lougheed's personal files); *see also* Lougheed Dep. Tr. at 244-245 (authenticating and discussing the DECnet Phase V document).

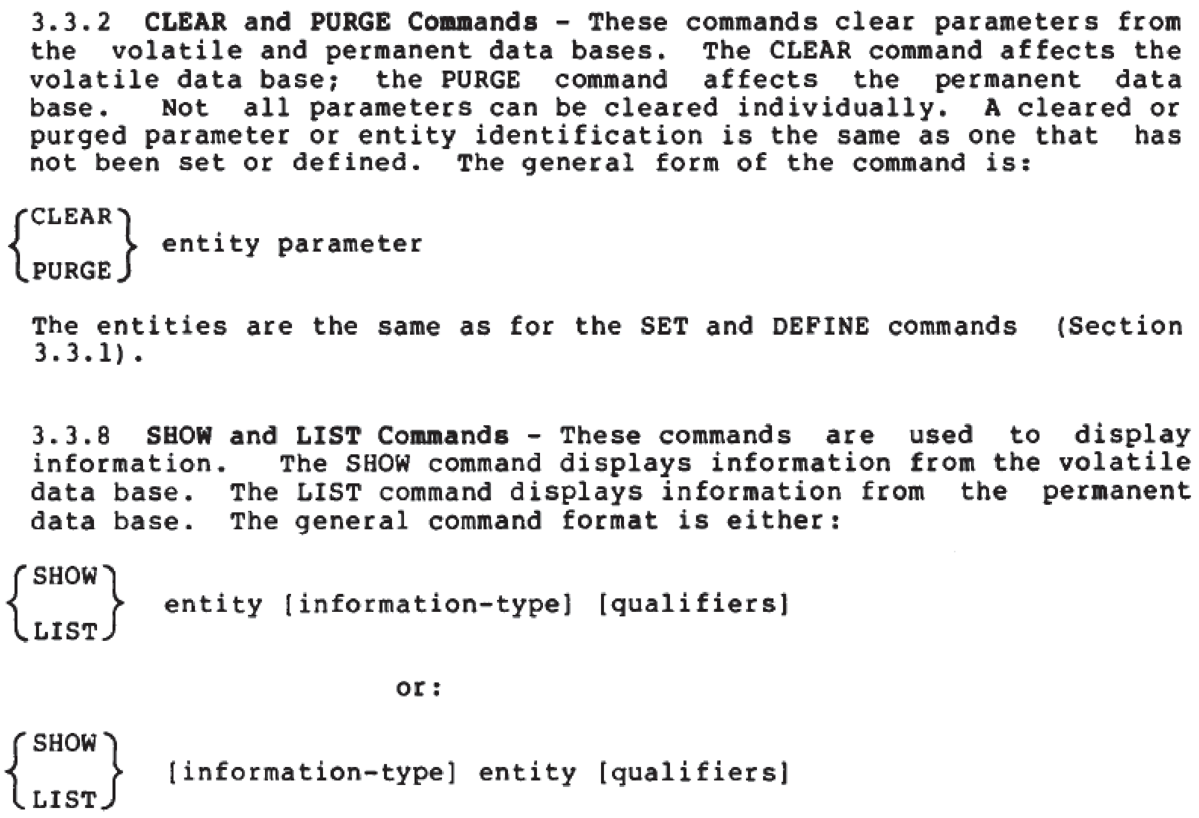
564. The DECnet Digital Network Architecture, at least as of 1980, featured a Network Control Program (NCP) that used a CLI with a multi-word command syntax very similar to those at issue in this case:



```
All NCP commands have the following common syntax:

command  entity  parameter-option(s)

where:

command                    Specifies the operation to be performed, such
                           as SHOW or LOAD.

entity                     Specifies the entity (component) to which the
                           operation  applies,  such  as  LINE  or KNOWN
                           NODES.

parameter-                 Qualifies  the  command  by providing further
option(s)                  specific information.
```

ARISTANDCA13228703 at Page 15 (DECnet DIGITAL Network Architecture (Phase III) Network Management Functional Specification Order No. AA-K181 A-TK Version 2.0.0 October 1980).

565. The NCP CLI, at least as of 1980, also supported "clear" and "show" commands:



3.3.2 **CLEAR and PURGE Commands** - These commands clear parameters from the volatile and permanent data bases. The CLEAR command affects the volatile data base; the PURGE command affects the permanent data base. Not all parameters can be cleared individually. A cleared or purged parameter or entity identification is the same as one that has not been set or defined. The general form of the command is:

```
{CLEAR}
{PURGE}  entity parameter
```

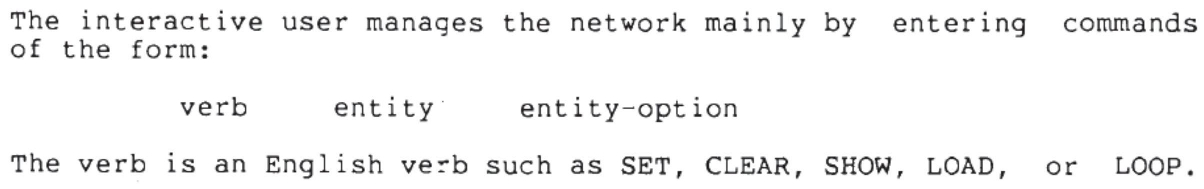
The entities are the same as for the SET and DEFINE commands (Section 3.3.1).

3.3.8 **SHOW and LIST Commands** - These commands are used to display information. The SHOW command displays information from the volatile data base. The LIST command displays information from the permanent data base. The general command format is either:

```
{SHOW}
{LIST}   entity [information-type] [qualifiers]

                      or:

{SHOW}
{LIST}   [information-type] entity [qualifiers]
```

ARISTANDCA13228703 at Pages 29-35 (DECnet DIGITAL Network Architecture (Phase III) Network Management Functional Specification Order No. AA-K181 A-TK Version 2.0.0 October 1980).

566. This DECnet NCP CLI command format persistent in Phase IV as well:



The interactive user manages the network mainly by entering commands of the form:

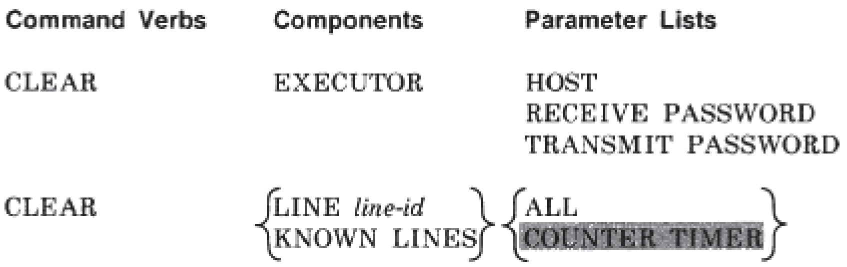
```
          verb      entity      entity-option
```

The verb is an English verb such as SET, CLEAR, SHOW, LOAD, or LOOP.

See ARISTANDCA13229492 at Page 15 (DECnet Digital Network Architecture Phase IV Network Management Functional Specification Order No. AA-X437A-TK Dec. 1983).

567. Finally, DEC also released DECnet-RSX, which refers to multiple

DECnet products. *See* ARISTANDCA13230687 (DECnet-RSX Guide to Network Management Utilities Order No. AA-EB30A-TC September 1985) (listing those products on Page Preface-1). The networking management utilities for the DECnet-RSX products also supported CLI commands that followed the familiar "verb component(s) parameter(s)" format, and specifically supported "clear" and "show" commands:

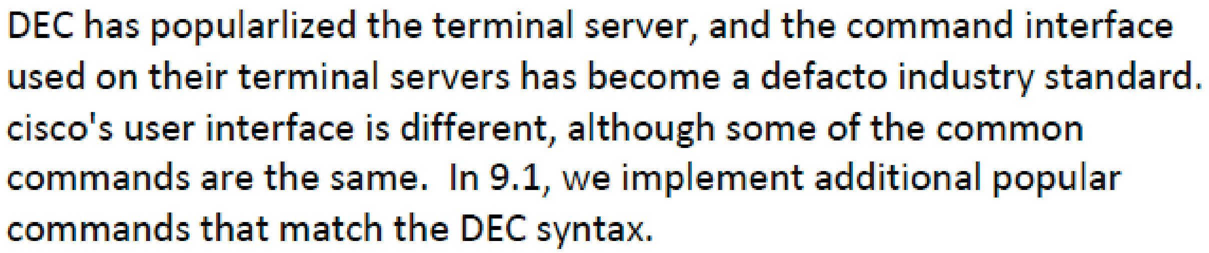


Command Verbs	Components	Parameter Lists
CLEAR	EXECUTOR	HOST RECEIVE PASSWORD TRANSMIT PASSWORD
CLEAR	LINE *line-id* KNOWN LINES	ALL COUNTER TIMER

Id. at pp. 1-6; *see also id.* at pp. 1-166 to 1-171 (showing the syntax for six different "show" commands, including "show line," "show system," and others).

568. Cisco's own documents acknowledge the Cisco IOS CLI's roots in these early DEC systems. For example, in an email discussion between Cisco employees regarding the context-sensitive help system in Cisco IOS, Cisco employee Aaron Leonard commented: "I've been using the '?' since TOPS-20 version 4 (20+ years ago), so at this point I'm probably neurologically incapable of doing anything else." *See* CSI-CLI-00810004 (Cisco email dated Nov. 16, 2005).

569. Another Cisco document from May 1994 similarly acknowledges:



DEC has popularlized the terminal server, and the command interface used on their terminal servers has become a defacto industry standard. cisco's user interface is different, although some of the common commands are the same. In 9.1, we implement additional popular commands that match the DEC syntax.

CSI-CLI-04978736.

570. Anthony Li, who worked at Cisco for many years, also testified that “the look and feel of the Cisco CLI had already been set [and] [t]he model was very clear from TOPS-20.” Li Dep. Tr. at 154. Mr. Li further explained, based on his experience working at Cisco, that while the source code was different between the Cisco IOS CLI and the TOPS-20 CLI, the way the Cisco IOS CLI functioned was copied from TOPS-20:

> **Q.** What were the similarities between the Cisco user interface and TOPS-20? ...
>
> THE WITNESS: So the Cisco user interface was taken -- the model was TOPS-20, so many of the external functionality in the CLI was similar to TOPS-20. The specifics about the syntax and the content were obviously somewhat different, and the implementation was completely different. ...
>
> **Q.** When you say “the implementation,” are you -- what are you referring to?
>
> **A.** So the TOPS-20 implementation, the code for that was written in a machine-specific language and had very specific set of code. We did not attempt to copy, reengineer or duplicate that in any way, shape or form. ***We simply copied some of the functionality.***
>
> **Q.** … So when you're referring to the implementation being completely different, you’re referring to the

> underlying source code; is that correct?
>
> **A.** Yes. The source code was completely different.

Id. at 145-146 (emphasis added).

571. Several disputed commands, like "show users" and "terminal length," were also supported by legacy DEC products. *See* ARISTANDCA00036076 at Page 20 (DECServer 2000 User Guide, showing support of "show users"); ARISTANDCA00038298 at Page 366 (TOPS-20 User Manual, showing support of the"terminal length" command).

572. The command keyword "enable" was also supported by DEC TOPS-20 before it was used in Cisco IOS. *See* ARISTANDCA00038298 at Page 137 (TOPS-20 User Manual, showing support of the "enable" command keyword to "activate[] capabilities").

573. MS-DOS was another operating system that pre-dated Cisco IOS in the early 1980s, and was known to early Cisco engineers. It too featured a command-line interface, and had a user prompt with the ">" (closed angle bracket) character. *See, e.g.*, Lougheed Dep. Tr. at 121.

574. There were also print controlling programs that existed before the Cisco IOS CLI was built, including the Quasar print controlling program, which was used in the EECF department at Stanford when Mr. Lougheed worked there in the early 1980s before he became an employee at Cisco. *See* Lougheed Dep. Tr. at 260. Quasar had a user interface program called OPR, which had a CLI. *Id.*[112] That CLI used commands that started with the first word "show," such as "show route-table." *Id.*; *see also* KL-

[112] As Mr. Lougheed noted at his deposition, in the early 1980s, there were no Graphical User Interfaces available, and therefore every system used a CLI. *See* Lougheed Dep. Tr. at 260.

00001699 (email from Mr. Lougheed's files showing his personal use of a Quasar OPR CLI "show" command).

575. I understand that Cisco, in its discovery responses, claims that Mr. Lougheed is the original "author" of the disputed command modes, associated prompts, and command hierarchies in the Cisco IOS CLI. *See* Cisco's Responses to Interrogatory Nos. 5 and 16. I further understand that Cisco claims that Mr. Lougheed is the "author" of several of the asserted CLI commands . *See* Exhibit F to Cisco's Responses to Arista's Interrogatories (including all supplements to Exhibit F).

576. Mr. Lougheed admitted at his deposition, when presented with his Stanford EECF (Electrical Engineering Computer Facility) employee personnel files and his own representations of his purported technical qualifications in his resume and job applications, that he was already experienced with the Digital Equipment Corporation TOPS-20 operating system as of 1980, and DECSYSTEM-20 mainframes (including DECSYSTEM-2060 systems) as of 1983. *See, e.g.*, Lougheed Dep. Tr. at 212-222 and exhibits discussed therein; KL-00000872. The DECSYSTEM-20 mainframes that Mr. Lougheed worked with at Stanford ran the TOPS-20 operating system. Lougheed Dep. Tr. at 226.

577. Indeed, Mr. Lougheed was so familiar with TOPS-20 as of 1983 that he served as the Stanford contact with DEC for field testing of new releases of the TOPS-20 operating system. Lougheed Dep. Tr. at 226. He also had TOPS-20 user documentation in his possession while an employee at Stanford, and before he began working at Cisco. *Id.* at 242-243 and associated exhibits (admitting that he had possession while at Stanford of TOPS-20 user documentation beyond the manuals he produced in this lawsuit).

578. Mr. Lougheed's extensive experience with the TOPS-20 CLI functionality influenced the CLI functionality he purportedly added to the Cisco IOS software. For example, Mr. Lougheed admitted at his deposition that the choice of an "EXEC" command mode in Cisco IOS was "inspired by the TOPS-20 command processor" and was therefore not his original creation. *See* Lougheed Dep. Tr. at 109. And he further admitted that calling a command mode "privileged" did not originate with Cisco. *Id.* at 112 ("**Q.** Did Cisco come up with the nomenclature of calling a mode 'privileged', to your knowledge? **A.** I don't believe — I don't believe Cisco came up with that terminology. "). He also admits that he "adopted" the convention of using a "percent sign leading a message [to] indicate[] that you are looking at an error message" directly from TOPS-20. *Id.* at 123-124; *see also* CSI-ANI-00043306 (Mr. Lougheed's email containing that admission).

579. Mr. Lougheed also confirmed at his deposition that his Stanford EECF team had full responsibility for multiple Digital Equipment Corporation VAX-11/780 and VAX-11/750 super-minicomputers and minicomputers, and that at least one member of his team, including Mr. Gregory Satz (who was also deposed in this lawsuit and an early employee at Cisco), worked directly with those Stanford EECF VAX systems. *See, e.g.*, Lougheed Dep. Tr. at 222-224. As confirmed by Mr. Lougheed, the DEC VAX systems used by the Stanford EECF department as of 1983 ran operating systems called VAX/VMS and Berkeley UNIX. *Id.* at 224.

580. In addition to the foregoing systems, as discussed above in the section of this Report addressing the origins of Cisco's routing software in Stanford EECF and SUMEX source code, several features and functionality of the Cisco IOS CLI were also

copied by Mr. Lougheed from routing systems developed at Stanford and written by Stanford employees including William Yeager. Specifically, the privileged and non-privileged command modes and their associated prompts (# and >) were already being used by Mr. Yeager's SUMEX software at Stanford, and Mr. Lougheed admitted that he used and modified Mr. Yeager's SUMEX software source code to generate the first Cisco routing software. Lougheed Dep. Tr. at 363-371. And Mr. Yeager's SUMEX software already supported what Mr. Lougheed admits are "common" CLI parser features like command completion, unique command detection (based on entering partial commands), and context-sensitive help triggered by the "?" character. Lougheed Dep. Tr. at 233-235.

581. Moreover, Mr. Lougheed confirmed in his deposition that he was already familiar with the use of "show" commands in non-Cisco systems that predated the Cisco IOS. For example, Mr. Lougheed testified that the Quasar print controlling program at Stanford used "show" commands in its OPR CLI, and that he had to use the Quasar OPR CLI a lot to fix the printers at Stanford. *See* Lougheed Dep. Tr. at 260; KL-00001699. Mr. Lougheed had therefore already used "show" commands frequently as a Stanford employee on the Quasar system. Finally Mr. Lougheed admitted at his deposition that he was aware that MS-DOS, which was released in the early 1980s before Cisco existed, used a ">" character as a command prompt. *See, e.g.*, Lougheed Dep. Tr. at 121.

582. In sum, Cisco engineers responsible for creating the Cisco IOS routing software were well aware of and had experience with legacy CLI functionality, and copied those features when creating the Cisco IOS CLI.

3. Terms used in the accused CLI commands come directly from industry standards, and well-known, descriptive industry parlance.

583. Based on my review of the asserted commands, the relevant industry standards and industry publications (many of which are discussed in detail in **Appendices A and B**), and my knowledge and expertise in the networking industry, it is my opinion that the vast majority of terms used in the accused CLI commands come directly from industry standards, and well-known descriptive industry parlance.

584. As discussed in **Appendices A and B**,which I incorporate into this section of my Report, the vast majority of command keywords used in the asserted CLI commands (and command fragments) are found in, defined by, and used throughout industry publications and standards from standards-setting bodies like the IETF and the IEEE. These publications are frequently the product of multiple vendors (typically part of a subject-specific Working Group) contributing ideas and collaborating for the sake of promoting interoperability between networking devices. The use in CLI commands of terminology found in those IETF and IEEE standards and publications that relate specifically to the functionality provided by each such command is expected and important to users of networking equipment CLIs.

585. For example, former Cisco employee Tony Li, who also worked for several competitors of Cisco including Juniper Networks and Procket Networks, made clear at his deposition that standardization is important to the networking industry, and that the use of industry standard-terminology was important to the development of Cisco CLI commands, simply because it minimized confusion:

> **Q.** Mr. Li, do you believe standardization is important

> to the networking industry?
>
> **A.** Extremely important. Without standardization we do not get to interoperability; without interoperability, we don't have a network.
>
> **Q.** To what extent, if at all, was the use of industry-standard terminology encouraged in Cisco's command sets?
>
> **THE WITNESS:** So the use of industry-standard terminology was encouraged largely by the engineering community of which much of Cisco's development was driven by simply because it minimized confusion.

Li Dep. Tr. at 129-130.

586. Moreover, the use of industry-standard terms and phrases in CLI commands stems directly from the fact that engineers adding such commands typically reviewed the applicable IETF and/or IEEE standards and publications to familiarize themselves with both the functionality and terminology associated with the command. Several Cisco witnesses testified regarding the development procedures they followed at Cisco when new CLI commands and networking functionality were added to the Cisco IOS CLI (including NX-OS and other flavors of Cisco IOS in dispute).

587. For example, former Cisco engineer Tong Liu--who Cisco identified as the author of several CLI commands relating to industry-standard PTP (Precision Time Protocol) functionality--testified that she reviewed the IEEE PTP industry standard from

cover to cover at the start of the development process, and consulted those standards when adding new CLI commands to Cisco IOS:

> **Q**. When was the first time that you saw the IEEE PTP standard?
>
> **A.** That's when I was working on this industrial Ethernet switch development around 2008, I think. …
>
> **Q**. … So did you see the IEEE PTP standard before you began adding PTP functionality to the Cisco industrial Ethernet switch?
>
> **A.** When you say "before." it's before I started writing code?
>
> **Q.** Yes.
>
> **A**. I — yes, I read the spec … — for understanding — to understand how it works. …
>
> **Q.** And did you read the entire standard before you began working on the PTP functionality?
>
> **A.** Yeah, I believe I read the -- the entire -- or the majority part of it.
>
> **Q.** That's -- that's impressive. How -- the standard is -- is several hundred pages long. But you read the whole thing -- you remember reading the whole thing?
>
> **A.** Yes.

Liu Dep Tr. at 94-97; Ex. 93 (IEEE Standard 1588-2008, Precision Time Protocol, ARISTANDCA00031733).

588. Ms. Liu also confirmed that she consulted the IEEE 1588-2008 PTP standard while she was coding the PTP functionality for Cisco's switches because "[a]ll of the messages format, the field definitions behaviors, are documented here [in the standard]." Liu Dep. Tr. at 97. She also confirmed that the PTP IEEE standard defines terms and acronyms, as well as functionality, that she ultimately implemented in Cisco switches. *Id.* at 105-118 (also confirming that the defined acronym "ptp" in the IEEE standard refers to the same "ptp" command keyword in the disputed PTP-related CLI commands, that the standard defines and uses PTP-relevant terms like "clock," "parent clock," "parent," and that certain mandatory parameters like "priority 1" and "priority 2" are defined by the IEEE standard); *id.* at 121-122 (providing similar testimony regarding the PTP standard's use and definition of "sync interval").

589. The disputed CLI commands she added to the Cisco IOS CLI include industry-standard acronyms and terms that are expressly defined by the IEEE industry standard. This includes "ptp" as the defined acronym for "Precision Time Protocol" and several "mandatory" (in the IEEE standards documents, a "MUST" support feature required for compliance with the standard) parameters and attributes for PTP, including the "priority1" and "priority2" parameters. When asked about the meaning of the terms in her various PTP commands, Ms. Liu confirmed at deposition that they meant that same thing--and were intended to carry the same meaning--as the corresponding terms in the IEEE PTP standard 1588-8000:

> **Q.** Now, does the priority 1 parameter in the CLI command "PTP priority 1," does that refer to the

> priority 1 attribute in the IEEE standard marked as Exhibit 93?
>
> … **THE WITNESS:** Yes. I think I chose it for the intention to mean the priority 1 attribute of the clock.
>
> … **Q.** And is your answer the same for the command "PTP priority 2"? Is the priority 2 command parameter -- does that refer to the priority 2 attribute in the IEEE standard marked as Exhibit 93?
>
> … **THE WITNESS:** It's referring to the same - that attribute, yes.
>
> **Q.** That attribute in the IEEE standard?
>
> **A.** In the IEEE standard, yes.
>
> **Q.** **…** And you knew about the priority 1 and priority 2 attributes in the IEEE standard before you started adding the "PTP priority 1" and "PTP priority 2" commands to the iOS software; correct?
>
> **A.** Yes, I read the spec.
>
> **Q.** And you were aware of those two particular attributes before you started adding the "PTP priority 1" and "PTP priority 2" commands to Cisco's routing software; right?
>
> **A.** Yes

See, *e.g.*, Liu Dep. Tr. at 137-141, 146-148 (providing similar testimony for these and other PTP command words like “sync interval”); 153-156 (same testimony for the PTP term “clock” and “parent” in the PTP-related CLI commands).

590. Former Cisco engineer Devadas Patil provided similar testimony at his deposition. Mr. Patil, who Cisco identified as the “author” of several CLI commands relating to LLDP (Link Layer Discovery Protocol, an industry standard protocol defined by the IETF), described in detail a multi-stage process for the development of new Cisco IOS functionality where the review and analysis of any related industry standards precede the addition of new Cisco IOS CLI commands. Patil Dep. Tr. at 80-82 (testifying that new CLI commands and the syntax for them are part of Stage 3 of a five stage process); 115-117 (confirming that LLDP is a ratified industry standard, that he had no role in its standardization, but that as soon as he was tasked with the LLDP project in late 2005, he “researched it actively and wanted to know as much of it as possible as early as possible.”).

591. Mr. Patil explained that the development process for implementing LLDP in the Cisco IOS software proceeded in stages: (1) market analysis/requirements gathering; (2) architecture; (3) design; (4) implementation; and (5) testing. Patil Dep. Tr. at 70-71, 124-125. As part of that development process, he looked at other vendors’ offerings in the LLDP space (as part of Stage 1), including discussions with HP about their ProCurve implementations of the LLDP industry standard. *Id.* at 72-78.

592. Critically, Mr. Patil also confirmed that he reviewed the IEEE standards documents (802.1AB) for LLDP in the 2005 time period as part of “Phase 1” of the multi-stage development process. Patil Dep. Tr. at 116-120 (confirming that Ex. 315 at his deposition was the IEEE standard for “802.1AB, which is the technical name for

LLDP"); Ex. 315 (IEEE Standard 802.1AB) (ARISTANDCA00017907). He further testified that he made a "full attempt" to review the IEEE LLDP standard from front to back in order to do "a very solid job of the architecture." *Id.* at 130-131. This is because Cisco's implementation of LLDP, unsurprisingly, was based on the 802.1AB standard. *Id.* at 144-145.

593. Mr. Patil also confirmed that several terms used in the LLDP commands were widely used in the LLDP standard, such as TLV. Patil Dep. Tr. at 76. Indeed, Mr. Patil readily admitted that he had become familiar with, or was already familiar with, many terms and acronyms defined or used in the IEEE 802.1AB (LLDP) standard, including "LLDP" and "TLV" and "neighbors," based on his careful review of the LLDP standards at the start of the development process. *Id.* at 148-153.

594. In terms of required LLDP functionality, Mr. Patil also confirmed that several of the LLDP-related CLI commands disputed in this lawsuit relate to mandatory LLDP functionality described in the IEEE 802.1AB standard. *Id.* at 155-156 (discussing the requirement that LLDP-compatible devices support "transmit-only" and "receive-only" and "transmit and receive" modes, and confirming that Cisco's implementation supported those modes); *see also id.* at 157-160 (discussing other mandatory LLDP features implemented in Cisco IOS). Not surprisingly, the LLDP commands at issue in this litigation use the same terms that the LLDP standard uses to describe such standardized functionality. This is because, as Mr. Patil confirmed, he tried to use command names that were familiar to people in the industry as well as accepted industry acronyms:

> **Q.** When you came up with the commands listed on

> Exhibit 316 [listing of Mr. Patil's commands for which Cisco claimed he was the "author"], did you try to pick names that would be familiar to people in the industry?
>
> … **THE WITNESS:** Yes.
>
> … **Q.** And did you try to use accepted industry acronyms when coming up with the commands listed in Exhibit 316?
>
> … **THE WITNESS:** Yes.
>
> … **Q.** Did you consider the vocabulary of the intended user of the LLDP functionality when you were coming up with the commands listed on Exhibit 316?
>
> … **THE WITNESS:** Yes.

Id. at 170-172.

595. Mr. Patil also confirmed that the development process for implementing LLDP functionality--at least the stages that he was personally involved in--lasted at least three years, and that the process for coming up with new command syntaxes took "maybe three man days to four man days." Patil Dep. Tr. at 62-64, 129-130. For many of his LLDP commands, he testified that coming up with the command syntax took "15 minutes." *Id.* at 188-197.

596. Abhay Roy provided similar testimony regarding several OSPF-related commands, including confirming that many of the words in the CLI commands he added to Cisco IOS came directly from the OSPF IETF RFCs and other RFCs relating to IP and BFD (including well-known and/or standards-defined terms and acronyms like "area,"

"authentication," "ip," "ipv6," "cost," "hello interval," " deadinterval" etc.). *See, e.g.*, Roy Dep. Tr. at 66-108; *see also generally* Roy Dep. Tr. (discussing the general industry usage and use in standards of words used in all CLI command attributed to him).

597. Mr. Roy also confirmed that he would re-use earlier command syntaxes for "ip" commands when adding new "ipv6" commands, noting "So we looked at -- we looked at what is existing in—in Cisco IOS implementation, and that, generally, is one of the overriding things; that don't reinvent the wheel. If there is something which is done, go with it." Roy Dep. Tr. at 107.

598. Former Cisco engineer Ram Kavasseri also testified at length regarding the SNMP-related commands he added to the Cisco IOS CLI. He noted that his team "was encouraged to participate in the IETF to define use standards around SNMP and network management." *See* Kavasseri Dep. Tr. at 46-48 (also confirming that SNMP is in fact an industry standard defined by the IETF). He also admitted that SNMP has been standardized by the IETF before he even joined Cisco as an engineer, that as a Cisco engineer he read the relevant SNMP RFCs, that he had no idea who came up with the term "SNMP," and that a wide variety of networking vendors used the term "SNMP" during his tenure at Cisco. *Id.* at 48, 53-55, 71-72.[113] Importantly, Mr. Kavasseri admitted with respect to feature development in Cisco IOS that "[i]f the feature had anything specific to do with an IETF document, then yes, I would have had to review the document to make sure I was implementing it correctly[.]" *Id.* at 62-63. It is therefore not surprising that many of the SNMP-related CLI commands in dispute use terms that

[113] I note that Mr. Kavasseri observed that while the IETF may define a protocol like SNMP, it becomes "industry standard only after companies pick it up and support it." Kavasseri Dep. Tr. at 48. In other words, even if formally defined, whether a protocol is actually an industry standard depends on its adoption by multiple companies. *See id.*

are part of, and used by, the SNMP protocol and the associated IETF documents. *See generally id.*; *see also id.* at 148-172 (discussing how several command keywords associated with Mr. Kavasseri's SNMP commands are found in SNMP IETF documents).

599. I note that the examples I describe here are only some of the testimony admitting to the fact that CLI authors typically follow standards documents in choosing command names, and review such documents before implementing any standards-related functionality. The full list of the testimony I have reviewed and relied upon for this purpose is as follows: Individual and Corporate Depositions of Adam Sweeney; Kenneth Duda; Hugh Holbrook; Tong Liu; Anthony Li; Abhay Roy; Ram Kavasseri; Phillip Remaker, Kirk Lougheed, Pradeep Kathail, Devadas Patil, Greg Satz, Terry Slattery, Mark Berly, Charles Giancarlo, Doug Gourlay, Lorenz Redlefsen, Sean Hafeez, Andre Pech, and Jayshree Ullal.

600. The consistency of this testimony from several Cisco engineers illustrates that the addition of new CLI commands for functionality that relates to an industry standard will be preceded by a careful review of those industry standard documents (whether an IEEE or an RFC) so that the engineer can familiarize himself or herself with the relevant networking terminology, the functionality required to comply with the standard, and the terms and acronyms that may be defined by the standard.

601. Cisco's corporate witness on CLI design also confirmed that Cisco engineers have a variety of resources to consult when adding new CLI commands, including industry standards from the IEEE and IETF, as well as customer feedback and the existed command set that may already be supported in the Cisco IOS CLI at the time the new command is being added. *See* Remaker Dep. Tr. at 154-157. Cisco's corporate

witness on CLI design, however, did not know what any of the "authors" of any of the asserted "command expressions" actually reviewed and considered to come up with the accused commands, and could not refute the standard practice of using standards terms whenever possible. *Id.* at 130-150.

602. A "best practices" document called the Parser Police Manifesto also set forth guidelines regarding the constraints placed on the addition of new CLI commands. *See*, *e.g.*, Remaker Dep. Tr. at 82-84 (identifying version 6 of the Cisco Parser Police Manifesto as the current version of the document as of the date of the deposition), Exh. 438 (Cisco Parser Police Manifesto, Version 6, CSI-CLI-04824213); *id.* at 40-43 (discussing the 1999 version of the Parser Police Manifesto), Exh. 436 (Jan. 1999 version of the Cisco Parser Police Manifesto, CSI-CLI-00754391).

603. The Cisco Parser Police Manifesto, which has existed at Cisco since at least the late 1990s (*see* Remaker Dep. Tr. at 29), places several practical and technical constraints on the addition of new CLI commands in Cisco IOS, including:

○ Select commands so that the parse tree remains extensible and "structure the parse tree not to have 'dead ends.'"

○ Use (or don't use) hyphens to ensure that parse chains remain extensible.

○ "Watch for collisions. Since the parser looks for smallest unique match, be on the lookout for adding an obscure keyword that conflicts with a common one."

○ "When naming a command, try to ***pick names that would be familiar to people in the industry***. For example, 'ip mtu 576' is better than 'ip maximum-transmission-unit 576" since MTU is an ***accepted industry acronym***."

○ "Do not use code names in commands. 'Debug whizzy-asic' or 'debug walamazoo' will not be very ***useful to customers***."

○ "***Commands should tend to be self-explanatory*** so that a relatively knowledgeable user can figure out the command function from the command and on-line help without having to scurry off to the manuals. What constitutes 'self-explanatory' will vary by your target audience, so be prepared to defend that point. While a non-ATM user may find the command 'forward-peak-cell-rate-clpl' offensively complex, the point can be made that this will be ***the only acceptable syntax for the ATM community based on the vocabulary and culture of that user group***."

See CSI-CLI-00754391 (Jan. 1999 version of the Parser Police Manifesto, CSI-CLI-00754391) (emphasis added).

604. As stated in the most recent version of the Parser Police manifesto, these constraints apply to all Cisco OSes that have a CLI interface. *See* CSI-CLI-04824213 (Cisco Parser Police Manifesto, Version 6); Cisco Corp. Dep. Tr. (Remaker) at 38-40 (testifying that most recent Parser Police Manifesto version made it clear that it applied to all Cisco CLIs).

605. Some of the constraints on the addition of new CLI commands are technical in nature and are tied directly to how the CLI parser operates. For example, a new CLI command cannot be the same as an existing CLI command. In other words, you

cannot have two networking functions tied to the exact same CLI command syntax, as the parser would not be able to determine which of the two functions the user intends to perform.

606. Similarly, the issue of "collisions" is one created by the technical workings of the CLI parser. The use of configuration and other CLI scripts by customers—where CLI commands, or even abbreviated CLI commands, are written out in a careful sequence of CLI commands in order to effect a particular configuration change to a networking device, or perform some other automated functionality on a networking device—nicely illustrates the problem of "collisions" when adding new CLI commands. A well-known short form command (or command abbreviation) like "sh int" for "show interfaces" will no longer work in both the CLI and in CLI scripts if an engineer adds a new command like "show internet" or "share interfaces" because the addition of such commands would render the abbreviation "sh int" ambiguous--and therefore invalid-- as a short form command. CLI scripts that might use the common "sh int" short form of "show interfaces" will then trigger an error and fail.

607. Cisco corporate witness Phillip Remaker testified at length about the importance of avoiding collisions and the impact on customers. *See* Remaker Dep. Tr. at 63-66 (discussing "collisions" and the example of breaking the "sh int" short-form command, and the "unfortunate consequences" it would have on customers).

608. The impact on customers and configuration scripts is also the reason why the Parser Police Manifesto discourages CLI command syntax changes: "***Changing an existing syntax is usually a bad idea***. Once customers are already using a certain syntax, changing the syntax will frequently do more harm than good. … ***[C]ustomers are trained***

on and familiar with existing syntax. … Having portions of the configuration be unrecognized could cause catastrophic failures." *See* CSI-CLI-00754391 (Jan. 1999 version of the Parser Police Manifesto, CSI-CLI-00754391) (emphasis added); *see also* Remaker Dep. Tr. at 75-79 (confirming the accuracy of these statements and stating "An engineer coming up with a new CLI command, as a best practice, should consider the potential impact on customer scripts.").

609. While I understand Cisco characterized these "constraints" as merely best practices and guidelines, Cisco witnesses repeatedly emphasized the importance of maintaining consistency and usability in the CLI, and in fact all user interfaces, given the purpose and function of a user interface. For example, Mr. Remaker--Cisco's corporate witness on topics relating to CLI command design--testified:

> **Q.** … Why do you believe that customers expect consistency of the configuration interface to Cisco IOS?
>
> **A**. I believe that that is an engineering guideline applicable to all user interfaces.
>
> **Q.** What is the consequence of having an inconsistent CLI?
>
> **A.** Customers will be unable to use Cisco products in the way that they expect.
>
> **Q.** Is ensuring consistency of the configuration interface to Cisco IOS an important consideration when creating new CLI commands?

> **A.** New CLI commands should be consistent with the design of the existing CLI.

Remaker Dep. Tr. at 45.

610. Mr. Remaker, testifying on behalf of Cisco, also explained the importance to the customers (users) of the CLI of using familiar industry terminology in command words:

> **Q.** … How does picking command words that would be familiar to people in the industry ensure consistency of the CLI? ...
>
> **THE WITNESS:** We believe that's what customers would expect.
>
> **Q.** And how does picking command words that would be familiar to people in the industry ensure usability of the command-line interface?
>
> **A.** We believe that customers would expect that.
>
> **Q.** Customers would expect the CLI commands to use words that would be familiar to people in the industry, correct?
>
> **A.** In the context of the command, yes.
>
> **Q.** When you say "in the context of the command," what do you mean by that?
>
> **A.** One of the attributes of "ip" is "mtu." So "mtu" is a reasonable term to pick related to "ip."

Remaker Dep. Tr. at 69-70.

611. For the same CLI usability reasons, Cisco's corporate witness, Mr. Remaker, confirmed that using obscure "code names" in new CLI commands is a bad practice because it"***makes the product less usable***." Remaker Dep. Tr. at 70-71 ("**Q.** How does using code names in CLI commands make Cisco IOS less usable? **A.** Customers would have no reason to know our internal code names."). This focus on usability clearly illustrates the functional nature of CLI commands, and the CLI as a whole, and also illustrates the external factors that constrain the selection of CLI commands.

612. Cisco engineer Pradeep Kathail further confirmed these facts at his deposition:

> **Q.** Can you explain to me what criteria you used when you came up with the words to be used in a CLI command? ...
>
> **THE WITNESS:** Most of the time you want to make sure that the words are such that they are self-explanatory, okay, as well as they are the terminology you are using [or "used to"] from the networking world.
>
> *********
>
> **Q.** And you mentioned when Mr. Silbert asked you about make sure that the words are such that they are self-explanatory, what did you mean by that?

> **A.** You want to use the words where customer doesn't have to go through and read a dictionary and say what does that word mean. So we want to make sure that they are coming from vocabulary which network users use day in and day out so you want to use very known familiar words.

Kathail Dep. Tr. (Rough) at 194 (including corrections of errors in rough transcript since final transcript was not yet available as of the date of this report).

613. I further note that the technical constraints placed on the addition of new CLI commands by virtue of how the CLI parser works is not a discretionary constraint. The creation of a "collision," the addition of a duplicate CLI command, or the creation of "dead ends" in a parse tree directly impacts the functionality of the CLI.

4. Cisco witnesses confirmed that they were already familiar with networking protocols and parlance before adding CLI commands.

614. As noted on several occasions above, Cisco witnesses also routinely confirmed their knowledge and familiarity with common hardware and protocols in the networking industry prior to working on any of the disputed CLI functionality.

615. For example, Mr. Lougheed confirmed at his deposition that he was already familiar with ethernet and ARPANET long before Cisco existed, and long before he purportedly created any aspect of the Cisco IOS CLI. *See* Lougheed Dep. Tr. at 226-227. Mr. Lougheed also repeatedly conceded at his deposition his familiarly with industry terms and phrases that were well known at the time he added CLI commands

that adopted those terms and phrases:

- ○ He was familiar with Address Resolution Protocol (ARP) long before he purportedly created any disputed IOS CLI features and functionality. Lougheed Dep. Tr. at 79-81, 226-227.
- ○ He was familiar with the term "flow control" in the networking industry. Lougheed Dep. Tr. at 239-240.
- ○ He was familiar with the term "spanning tree" by the late 1980s, well before any accused "spanning tree" commands were added to Cisco IOS, reviewed spanning-tree-related IEEE standards in the late 1980s, and didn't come up with the term "spanning tree." Lougheed Dep. Tr. at 251-252, 331-335.
- ○ He knew that the IEEE had been using the industry standard terms "mac," media access control," and "mac address" at the time he added the disputed "mac-address" command to the Cisco IOS CLI. Lougheed Dep. Tr. at 246-247, 319-321.
- ○ The BGP standardization process had already begun in 1989 with the publication of RFC 1105 in June 1989 by the time he added the "timers bgp" command to Cisco IOS. Lougheed Dep. Tr. at 345-346.
- ○ He was familiar with "RIP" (Routing Information Protocol) when it was an "informal standard" widely adopted in the industry, and used that well-known acronym in commands. Lougheed Dep. Tr. at 125-126.

- ○ He worked with an IBM employee, Mr. Rekhter, to standardize BGP, and used that acronym in BGP commands. Lougheed Dep. Tr. at 126 to 127.
- ○ He was already familiar with the common networking term “ip address” before he joined Cisco. Lougheed Dep. Tr. at 130-131.
- ○ He did not come up with the common networking terms “domain name” and “domain lookup” and was already familiar with them when adding those commands to Cisco IOS. Lougheed Dep. Tr. at 143-144.
- ○ He was familiar with the well-known industry acronym “MOTD” (message of the day) and did not come up with that acronym for the “banner motd” command. Lougheed Dep. Tr. at 175.
- ○ He was familiar with the well-known industry terms “boot” and “system” before adding the “boot system” command to Cisco IOS. Lougheed Dep. Tr. at 181.

616. Similarly, other Cisco witnesses (including former Cisco engineers) noted their familiarity with industry-standard terms based on prior experience with routing equipment, education, and simply reading the relevant industry-standard documentation before adding new CLI commands. *See* Discussion of Liu, Patil, Roy, and Kavasseri depositions above regarding their reliance on industry standard documents *prior to* adding CLI commands that relate to those standards; *see also* Anthony Li Dep. Tr. at 13-56 (discussing Mr. Li’s extensive knowledge of almost all of the routing protocols and standards at issue in this lawsuit, his training on Cisco equipment, and his familiarity with

legacy equipment including UNIX, TOPS-20, and VAX/VMS, which he gained ***prior to*** joining Cisco as an engineer). Indeed, Mr. Li--who added a large number of CLI commands at issue here--readily admitted throughout his deposition that technical terms used in industry standards (including the acronyms for the standards themselves) were well-known to the networking community that used those standards. *See id.*; *see also id.* at 37, 38, 48, 52-55, 107.

617. Importantly, Mr. Li and others freely acknowledged that they had no belief that the terms used in industry standards were proprietary to any one vendor, including Cisco. *See* Anthony Li Dep. Tr. at 109-110; *see also* Lougheed Dep. Tr. at 125-126 (stating that he did not belief “RIP” was proprietary because it was widely used across the industry even before it was described and standardized in an RFC).

618. These examples, and the others cited above, demonstrate that Cisco engineers routinely familiarized themselves with the terminology of relevant networking protocols and common industry parlance before selecting CLI commands, and would use that terminology when adding new CLI commands.

5. Cisco witnesses confirmed that they would copy existing CLI command syntax when adding new commands.

619. Many of the CLI “command expressions” disputed in this litigation are substantially the same or similar to other “command expressions” that already existed in the Cisco IOS CLI. Clear examples of this are commands that relate to functionality of newer versions of a routing protocol--for example, IPv6 and OSPF version 3--where the commands for the newer versions of the routing protocol mirrored the pre-existing commands for the older version of the protocol.

620. Cisco's corporate witness on CLI command creation, Mr. Remaker, confirmed that Cisco engineers would be encouraged to reuse existing command syntaxes already present in the Cisco CLI when adding new CLI commands to ensure "product consistency." Remaker Dep. Tr. at 150-151. For example, new CLI commands for OSPF version 3 functionality were merely "copies of the corresponding ospfv2 commands." *See* CSI-CLI-00608702.

621. This was also echoed by Mr. Patil (who Cisco identified as the "author" of several LLDP-related CLI commands). When confronted with a document that showed that Mr. Patil had copied the syntax from existing CLI commands for a new command, Mr. Patil testified:

> **Q.** Do you think a user's familiarity with an existing command set is important to consider when adding new commands to a CLI?
>
> .... **THE WITNESS:** I consider that important, yes.
>
> **Q.** And, in fact, in Exhibit 319 here, you use an identical command syntax for something that was already in the CLI for a new command; correct?
>
> **A.** Yes, but it was on a different construct.
>
> **Q.** Sure, but in terms of the command syntax -- and we are using the command syntax -- that was an intentional choice on your part; correct?
>
> **… THE WITNESS:** Yes.
>
> … **Q.** And that was because the users would already

> been familiar with the existing command sets in the CLI; correct?
>
> … **THE WITNESS**: Yes, consistency -- the answer is yes.

Patil Dep. Tr. at 179-180; Ex. 319 (E-mail dated 10/10/07 from Devadas Patil) (CSI-CLI-00836482). Mr. Patil repeatedly confirmed this practice of copying the syntax of existing commands when adding new LLDP commands, where he followed the "exact same pattern as the ones [CLI commands] that were approved on parser-police for basic lldp." Patil Dep. Tr. at 181-182; Exh. 320 (CSI-CLI-00817320). Again, Mr. Patil testified that he did this "for the sake of consistency and skills transfer" for the user. Patil Dep. Tr.. at 182-183.

622. Mr. Patil also confirmed that he tried to copy the same command syntax for new LLDP commands as Cisco used for already existing CDP (Cisco Discovery Protocol) CLI commands. Patil Dep. Tr. at 209-210.

623. Perhaps most important, Mr. Patil confirmed that he did not even consider departing from the established command syntaxes already in Cisco IOS when adding new LLDP commands to Cisco IOS. Patil Dep. Tr. at 183 ("Did we consider doing a very different syntax? No.").

624. Similarly, where an existing "command hierarchy" already existed (for example, there were already commands in Cisco IOS that started with particular keywords), Cisco witnesses testified that they were constrained by those existing commands and command hierarchies already supported by the Cisco IOS CLI. In many cases, those prior, pre-existing Cisco IOS commands are ***not*** at issue in this lawsuit, and

Cisco has provided no evidence that those pre-existing commands were original or creative.

625. For example, Mr. Lougheed admitted at his deposition that after there were already commands using the "'IP' as the keyword indicating Internet protocol-related stuff, [he] would have felt constrained to use that as the leading keyword" in subsequent new commands. *See* Lougheed Dep. Tr. at 317-318. And Mr. Lougheed also confirmed that once "show" commands were introduced into Cisco IOS, he was constrained to use "show" for all subsequent commands that provided information. *See id.* at 326-327.

626. And Mr. Patil confirmed that when he added "show" and "clear" commands relating to LLDP functionality, he just reused "show" and "clear" in those commands because they were already being used by other CLI commands in Cisco IOS. Patil Dep. Tr. at 199-200.

627. Examples of such "re-used" command syntaxes can be gleaned from reviewing Cisco's asserted list of commands, although in certain cases, the "old" command that was later copied by Cisco's engineers is not asserted in this lawsuit. For example, "area default-cost", "area nssa", "area nssa default-information-originate", "area nssa translate type7 always", "area range", "area stub", "default-information originate", "default-metric", "log-adjacency-changes", "maximum-paths", "passive-interface", and "router-id" are all commands that are listed multiple times by Cisco for different versions of OSPF (and, in the case of "log-adjacency-changes", it is listed also for IS-IS.

628. Other clear examples of replicated commands include IPv6 commands,

such as "show ipv6 bgp" and "show ipv6 ospf" families of commands, which largely replicate the "show ip bgp" and "show ipv6 ospf" families of commands.

6. Cisco witnesses confirmed that they would spend a minimal amount of time on the syntax of new commands.

629. As noted earlier in this Report in my discussion of the development process followed by Cisco engineers, several Cisco witnesses confirmed that the time and effort spent on coming up with the syntax for new CLI commands was insubstantial compared to the planning and source code writing that was required to implement the functionality associated with the commands.

630. For example, Mr. Lougheed confirmed that coming up with the syntax for the CLI command "ip access-group" took ""not very long" and a matter of minutes, as compared to a day's worth of time writing source code for the underlying functionality. *See* Lougheed Dep. Tr. at 317-319. And Mr. Satz testified that took all of fifteen seconds to come up with CLI commands for functionality that he estimated took months to create. Satz Dep. Tr. at 76. Similarly, Mr. Patil confirmed that for a project that took months to complete, for many of his LLDP commands, he came up with the command syntax in "15 minutes." Patil Dep. Tr. at 188-197.

631. This further emphasizes the minimal (if any) creative effort behind the addition of CLI commands, particularly given the constraints on the selection of CLI commands--both functional and technical--discussed earlier in this Report with respect to the Parser Police Manifesto and the re-use of existing CLI command syntaxes.

7. The Asserted CLI Command Hierarchies are not original.

632. I have already discussed the purely functional nature of command hierarchies, and the testimony provided by Cisco witnesses regarding hierarchies from Mr. Remaker and Mr. Lougheed, in my discussion of why the CLI is a method of operation, which I incorporate by reference into this section.

633. That idea of organizing related commands in a logical tree-like structure was not original to Cisco. As explained earlier in this Report in my discussion of legacy systems (which I incorporate into this section), early Cisco engineers, including Mr. Lougheed, had extensive experience with and were exposed to legacy systems like UNIX and also systems manufactured by DEC (including TOPS-20 and VAX/VMS) that had CLIs that organized commands in a hierarchical way. Mr. Lougheed also worked directly at Stanford, prior to joining Cisco, with a print management program that used a hierarchical command syntax, including support for a "show" command hierarchy. *See* Lougheed Dep. Tr. at 260. Similarly, as discussed earlier in this Report, DEC systems as early as 1980 supported a structured, multi-word command syntax that followed a hierarchical "verb" "object" "parameter(s)" structure, including "clear" and "show" command families. *See* ARISTANDCA13228703 at Pages 29-35 (DECnet DIGITAL Network Architecture (Phase III) Network Management Functional Specification Order No. AA-K181 A-TK Version 2.0.0 October 1980); *see also* Anthony Li Dep. Tr. at 23-24 (discussing the structured syntax of TOPS-20 and VAX/VMS); *id.* at 145-146 ("So the Cisco user interface was taken -- the model was TOPS-20, so many of the external functionality in the CLI was similar to TOPS-20."); *id.* at 154 ("The infrastructure, the

look and feel of the Cisco CLI had already been set. The model was very clear from TOPS-20.”).

634. Indeed, Mr. Satz—who worked alongside Mr. Lougheed at Stanford before Mr. Lougheed joined Cisco—confirmed that the DEC equipment at Stanford followed a hierarchical structure, and discussed how “show” commands in the DEC NCP (Network Control Program) CLI are used in a hierarchy. *See* Satz Dep. Tr. at 39-50; Exh. 401 (“TOPS-20 DECnet-20 Programmers Guide and Operations Manual”).

635. Cisco’s organization of the disputed multi-word CLI commands into such a tree-like structure was therefore not original. Moreover, to the extent that individual Cisco engineers simply copied the existing syntactical structure of existing Cisco IOS commands (for example, simply re-used “show” as the first command word, or simply re-used the “show ip bgp” structure for a new command), such copying of an existing hierarchy is just as unoriginal and devoid of creativity as copying the command words themselves from prior commands.

8. The Asserted CLI Modes and Prompts are not original.

636. As discussed in detail above, Mr. Lougheed admitted that the EXEC and Privileged EXEC command modes in Cisco IOS did not originate from him (or Cisco), but instead came from legacy systems including TOPS-20 and UNIX, and also came from software written by a Stanford employee, William Yeager. My analysis of those pre-Cisco systems confirms that they supported those command modes, and that the idea of separating CLI users into a “privileged” mode (where they could access more commands) and a “non-privileged” mode (where they could not access “privileged” commands) was known to Cisco engineers before the Cisco IOS CLI was developed.

9. The Asserted CLI Command Responses are not original.

637. The majority of the information that Cisco claims as protected command responses also fails to pass the originality threshold. The snippets that Cisco cites are minimal descriptive phrases regarding switch features and functionality. The substance of the phrases derive from the functionality of the device and industry terminology used to define that functionality.

638. For example, former Cisco engineer Tong Liu confirmed that the "show" command outputs for PTP-related commands, for which Cisco identified Ms. Liu as an "author," contained system-specific real-time configuration and state information about PTP parameters and objects (like the PTP-defined "grandfather clock") that were defined by the IEEE 1588-8000 PTP standard, because that standardized information would be "meaningful to show to the user."[114]

639. Consider also the excerpts of IGMP multicast configuration state that Cisco claims.[115] Here Cisco cites such mundane phrases as "IGMP version" and "Robustness value" (neither of which Arista uses). The phrases Arista does use are similarly mundane: "startup query count" and "last member query response interval." But the concept of a "query" in IGMP is set forth in the relevant IETF RFC.[116]

640. Cisco documents also emphasize the importance of clear command outputs that use industry-standard terms and abbreviations: "Counters SHALL use industry-standard abbreviations based on RFCs and other specifications. When no industry-standard abbreviations are available, counters SHALL be human-readable.

[114] Liu Dep. Tr. at 167-172.

[115] 9/1/2015 Cisco Supp. Resp. to Int. 2, Exh. E, at 9.

[116] See generally, RFC 2236, RFC 3376, RFC 4604.

Interface counters SHALL be displayed in show interface or in command output separate from show controllers." *See* CSI-CLI-00779340 (Cisco engineer proposing a "counter manifesto" to the Parser Police).

641. I have reviewed all of Cisco's Exh. E, which purports to set forth the relevant asserted command responses, and with the exception of at most few descriptive words connecting industry standard terms together, I find that the purported copied command response excerpts are derived from industry standard materials.

C. Summary of Opinions re Originality

642. As described more fully above, substantial portions of the asserted CLI commands, hierarchies, modes, prompts, and responses were not created independently by a Cisco author. Instead, they were either copied directly from another work or they were derived from another work, or common industry parlance. **Appendices A and B**, as well as **Appendix G**, contain an additional assessment of all disputed command keywords that, in my opinion, are not original to Cisco. I address the remaining portion of the asserted protected expression in subsequent sections.

X. ANALYSIS OF *SCENES A FAIRE* DOCTRINES

643. I have also been asked to analyze whether the asserted copyright infringement claim advanced by Cisco implicates the so-called "*scenes a faire*" doctrine. In so doing, I have considered a number of potential factors that limit the choices a Cisco engineer would have faced in deciding what command keywords to use to invoke a given function. I understand that, in applying the *scenes a faire* doctrine to the context of computer software, one should consider the effect the following factors may have on the composition of asserted protected aspect of the "works": hardware standards and

mechanical specifications, software standards and compatibility requirements, computer manufacturing design standards, industry programming practices, and practices and demands of the industry being serviced. I understand that the purpose of this doctrine is to exclude from copyright protection expression whose creation flowed naturally from considerations external to the author's creativity.

644. I start with the fact that using a command line interface itself is not original to Cisco, so the fact that there are other potential means of controlling a network switch--such as a graphical user interface, or other programmatic interfaces--is not pertinent to whether the similarities between Cisco's and Arista's CLI aspects are *scenes a faire*. Cisco having chosen to follow the path of predecessor operating systems such as UNIX and TOPS-20 by using a CLI, the *scenes a faire* analysis should focus on what constraints limited each CLI author's choice of command words.

645. Within the context of a CLI for a networking switch, there are a host of external forces that severely constrain one's choice of words to use in the commands. An important indicator of what Cisco's CLI authors followed when arriving at commands is the Parser Police Manifesto described above.[117] It advises to use terms that are familiar in the industry, only use acronyms accepted in the industry, and urges that commands be self-explanatory to avoid frustrating users. It also cautions against commands that lead to "dead-ends" where it is impossible to extend a feature with a new sub-feature, and warns of "collisions" that can occur when the shortened "command-completion" version of one

[117] I understand that Arista sought, but Cisco successfully opposed, the deposition of each CLI author. As a result of Cisco's successful opposition , there is not direct evidence about the "creative" process for each command. I reserve rights to supplement this section with any further evidence that may come to my attention prior to trial.